REMARKS

This Amendment is filed in connection with a Request for Continued Examination and a Petition for a 1-month Extension of time in response to the Final Office Action dated Oct. 30, 2007 and the Advisory Action dated Jan. 28, 2007. The Applicant respectfully requests reconsideration in light of the below discussion. All objections and rejections are respectfully traversed.

Claims 1-20 are pending in the case.

Claims 1, 14 and 20 have been amended to better claim the invention.

No new claims have been added.

Request for Interview

The Applicant respectfully requests a telephonic interview to advance the prosecution of this case. The Applicant believes an interview will be most productive after the Examiner has had an opportunity to review this Amendment, but prior to the issue of the next Office Action. As the Applicant can not determine when the Examiner will have time to consider this Amendment, given PTO workload, the Applicant respectfully requests the Examiner contact the Applicant at 617-951-2500 when he reviews this Amendment so that a time convenient to the Examiner may be arranged for a telephonic interview.

Claim Rejections - 35 U.S.C. §102

At paragraphs 1-12 of the Final Office Action, claims 1-4, 7-15 and 18-20 were rejected under 35 U.S.C. §102(e) over Huang, U.S. Patent Publication No. 2002/0046271 (hereinafter "Huang").

The Applicant's claim 1, representative in part of the other rejected claims, sets forth:

1. In a plurality of intermediate network devices having a plurality of ports for forwarding network messages within a bridged network having a root, the plurality of intermediate network devices organized as a stack, each intermediate network device having a stack port for use in communicating with the other network devices of the stack, a method for efficiently transitioning the ports among a plurality of spanning tree protocol (STP) states, the method comprising the steps of:

executing the STP at each intermediate network device of the stack so as to assign the stack port of each device to either a Root Port Role or a Designated Port Role, and to assign a non-stack port at a single device of the stack to the Root Port Role;

transitioning the ports assigned to the Root Port Role and the Designated Port Role to a forwarding STP state;

designating all non-stack ports at the devices of the stack that provide connectivity to the root, other than the non-stack port assigned to the Root Port Role, ***as Alternate Stack Root Ports***;

transitioning the Alternate Stack Root Ports to a discarding STP state; and

in response to a failure at the non-stack port assigned to the Root Port Role, transitioning a selected one of the Alternate Stack Root Ports from the discarding STP state directly to the forwarding STP state, without transitioning the selected port through any intermediary STP states, so that the selected Alternate Stack Root Port assumes Root Port Role for the stack.

Huang discloses an architecture for configuring a number of switching nodes into a stack. *See* abstract. The switching nodes include stack ports and non-stack ports. "The STP states of stack ports are determined by the Topology Discovery protocol 102." *See* paragraph 0064. Huang's Topology Discovery protocol "develop[s] a complete topology map of the current stack topology" (*see* paragraph 0060) by sending special periodic topology advertisements throughout the stack. *See* paragraph 0049. In contrast, the STP states of non-stack ports are determined using conventional STP. Huang describe that "[t]he **standard STP**, with some minor modifications, is run at each of the switching nodes. The STP running at a switching node determines the STP states of its own non stack ports." *See* paragraph 0061 (emphasis added). In response to receiving BPDUs that prompt a topology change, "**standard STP** determines the root port based upon the receiving port ID and the corresponding path costs." *See* paragraph 0062 (emphasis added)

Huang does not mentions how exactly he transitions a blocked non-stack port to a forwarding state to become the root port. Huang simply refers to using "standard STP." It is well known that in conventional STP standardized in IEEE 802.1 (i.e. "standard STP") intermediately states are used when transitioning from a blocked state to a forwarding state, for example as is needed for a port to assume root port role. The well-known textbook *Interconnections Second Edition*: *Bridges, Routers, Switches and Internetworking Protocols,* by Radia Pearlman published Jan. 2000 describes at page 67:

> The 802.1 standard actually calls for two intermediate states. This arrangement is designed to minimize learning of incorrect station locations while the topology has not yet stabilized. To minimize unnecessary forwarded frame, the committee deemed it desirable not to allow a bridge to start forwarding until it has built up its leaned cache. During the initial portion of the intermediate state, the bridge does not learn station addresses; the during the latter part, the bridge still does not forward packets over that port, but it starts learning station locations on that port. In other words, the intermediate state is subdivided into two substates: *listening* and *learning*.
> (copy of pages 66-67 of textbook attached herewith)

The Applicant respectfully urges that Huang is silent concerning the Applicant's claimed assigning "***designating all non-stack ports at the devices of the stack that provide connectivity to the root… as Alternate Stack Root Ports***" and "***in response to a failure at the non-stack port assigned to the Root Port Role, transitioning a selected one of the Alternate Stack Root Ports from the discarding STP state directly to the forwarding STP state, without transitioning the selected port through any intermediary STP states, so that the selected Alternate Stack Root Port assumes Root Port Role for the stack.***"

The Applicant claims a special ***Alternate Stack Root Port*** designation that allows a non-stack port to be transitioned from the discarding STP state directly to the forwarding STP state, ***without transitioning the selected port through any intermediary STP*** __**states**__, such as the listening STP state or the learning STP state. In this manner, the Applicant allows an Alternate Stack Root Port in the blocked state to assume Root Port Role in the forwarding state much more rapidly than with prior techniques. Huang simply describes using standard STP to transition non stack ports. For example Huang states "**standard STP** determines the root port based upon the receiving port ID and the corresponding path costs" for non stack ports. *See* paragraph 0062 (emphasis added). There is no indication that Huang uses a special transitioning technique that avoids transitioning through intermediate states. Rather, it appears Huang suggests operation in the standard manner, as set forth in the IEEE 802.1 standard, which operates quite differently than what is claimed here.

Further, in the Advisory Action the Examiner suggests that Huang's "slaves" are somehow akin to the claimed "***Alternate Stack Root Port***" designation. The Applicant respectfully urges such likening is misplaced. As made clear in the claims, an "Alternate Stack Root Port" designation is a special designation that may be assigned to non-stack ports that enables rapid transitioning to the forwarding state. In sharp contrast, Huang "slaves" are network devices, i.e. whole switches that operate under the control of a master switch. For example, at paragraph 0059 Huang state, "[t]he elected master forwards the commands to the other stack switches (i.e., the slaves)." Huang further states in his claim 18 that "one of said internetworking devices of said stack ...is elected as a master, said master internetworking device communicates stack-wide configuration information to the remaining internetworking devices, which are slaves." Clearly, a "slave" switch is not akin to an "***Alternate Stack Root Port***" designation for a port.

Accordingly, the Applicant respectfully urges that Huang is legally insufficient to anticipate the present claims under 35 U.S.C. §102 because of the absence of the Applicant's claimed novel "***designating all non-stack ports at the devices of the stack that provide connectivity to the root... as Alternate Stack Root Ports***" and "***in response to a failure at the non-stack port assigned to the Root Port Role, transitioning a selected one of the Alternate Stack Root Ports from the discarding STP state directly to the for-***

warding STP state, without transitioning the selected port through any intermediary STP states, so that the selected Alternate Stack Root Port assumes Root Port Role for the stack."

Claim Rejections - 35 U.S.C. §103

At paragraphs 13-15 of the Final Office Action, claims 5, 6, 16 and 17 were rejected under 35 U.S.C. §103(a) over Huang in view of "Admitted Prior Art."

The Applicant notes that claims 5, 6, 16, and 17 are dependent claims that depend from independent claims believed to be allowable for at least the reasons discussed above. Accordingly, claims 5, 6, 16, and 17 are believed to be allowable at least due to such dependency, as well as for other independent reasons.

Should the Examiner believe telephonic contact would be helpful in the disposition of this Application, the Examiner is encouraged to call the undersigned attorney at (617) 951-2500.

In summary, all the independent claims are believed to be in condition for allowance and therefore all dependent claims that depend there from are believed to be in condition for allowance. The Applicant respectfully solicits favorable action.

Please charge any additional fee occasioned by this paper to our Deposit Account No. 03-1237.

Respectfully submitted,

[illegible]

James A. Blanchette
Reg. No. 51,477
CESARI AND MCKENNA, LLP
88 Black Falcon Avenue
Boston, MA 02210-2414
(617) 951-2500

3.4.2 Avoiding Temporary Loops

After a topological change (a new bridge or link coming up or a bridge or link failing), it will take some time for news of the event to spread throughout the topology. Until news of the changed topology has reached all bridges, the bridges will operate on inconsistent data. This situation has two possible outcomes (both of which may occur simultaneously in various portions of the network).

1. There may be temporary loss of connectivity if a bridge port that was off in the old topology hasn't yet found out that it needs to be on in the new topology.
2. There may be temporary loops in the topology if a bridge port that was on in the old topology hasn't yet found out that it needs to be off in the new topology.

Although it was stated earlier, it is worth repeating that loops with bridges are far more frightening than loops with routers for two reasons.

1. There is no hop count in the data link header, so with bridges, packets will loop indefinitely until the topology stabilizes.
2. Packets can proliferate with bridges because a bridge might forward the packet onto several LANs, and several bridges might pick up the packet when it is transmitted onto a LAN. In contrast, routers forward the packet in only one direction and specify the router to which the packet is being forwarded. Therefore, a loop with routers will not cause packet proliferation.

Temporary partitions in a bridged topology are better than temporary loops. (Almost anything is better in a bridged topology than a loop.) You can minimize the probability of temporary loops by requiring that a bridge wait some amount of time before allowing a bridge port that was in the blocking state to transition to the forwarding state. Hopefully, the amount of time is sufficient for news of the new topology to spread. In that way, all bridge ports that should be off in the new topology have already heard the news and turned themselves off before any additional ports start forwarding data packets.

This timer should be at least twice the maximum transit time across the network. Suppose that bridge B1 is the root of the "old" topology. Suppose also that B1 issues one configuration message, which transits the network essentially instantaneously (no congestion). Then suppose that its next configuration message is maximally delayed by X sec. B1 then crashes. Bridges close to B1 will time out B1 and start to compute a "new" topology X sec before bridges maximally far from B1.

Now suppose that the root in the new topology is B2, which is maximally far from B1. And suppose that the first configuration message transmitted by B2 travels maximally slowly—that is, it takes X sec to reach the portion of the network that is near the ill-fated B1. Then bridges near B2 will find out about the new topology X sec before bridges that are close to B1.

In the worst case, news of the new topology will take twice time X to reach all bridges. Therefore, the algorithm does not allow a bridge to immediately transition a port from the blocking state to the forwarding state. Instead, the bridge temporarily puts the port into a different state: it still does not forward packets but does send Hello messages as if it were the Designated Bridge. After a timer expires, if the bridge still thinks the port should be in forwarding state, the port can be turned on. The purpose of this delay is to avoid adding extra connectivity until all bridges that might need to turn ports off in the new topology have had a chance to hear about the new topology.

The 802.1 standard actually calls for two intermediate states. This arrangement is designed to minimize learning of incorrect station locations while the topology has not yet stabilized. To minimize unnecessarily forwarded frames, the committee deemed it desirable not to allow a bridge to start forwarding until it has built up its learned cache. During the initial portion of the intermediate state, the bridge does not learn station addresses; then during the latter part, the bridge still does not forward packets over that port, but it does start learning station locations on that port. In other words, the intermediate state is subdivided into two substates: *listening* and *learning*.

Note: in the original spanning tree algorithm, I had only a single intermediate state, known as *preforwarding*. The committee asked whether bridges should learn station addresses in the preforwarding state. I said I didn't think it mattered. The committee decided to break preforwarding into the two aforementioned states. I believe that having two states is unnecessary and that bridges would work fine whether or not they learn station addresses in preforwarding. Breaking this period into two states makes the algorithm more complicated but does no harm. It would be an interesting project to study the trade-offs between the three strategies:

1. One intermediate state in which station learning is done
2. One intermediate state in which station learning is not done
3. Two intermediate states, as in the spec, in which learning is not done in the first half of the time and is done in the second half

3.4.3 Station Cache Timeout Values

Bridges learn and cache the location of stations. Because a station might move, it is important for a bridge to "forget" station locations unless it is frequently reassured that the learned information is correct. This is done by timing out entries that have not been recently verified.

Choosing a suitable timeout period is difficult. If an entry is incorrect for any reason, traffic may not be delivered to the station whose location is incorrectly cached. If an entry has been deleted, traffic for the deleted station will leak into other portions of the bridged network unnecessarily. If the timeout is too long, traffic will be lost for an unreasonably long time. If the timeout is too short, performance in the network will degrade because traffic will be forwarded unnecessarily.

If the only reason for a station's location to change is that the station moved, a timer on the order of minutes is reasonable for three reasons.